

Out of the Ivory Tower: Challenges and Opportunities in Technology Transfer

Tao Xie

Peking University (2011-2012), China
North Carolina State University
Raleigh, NC, USA

In Collaboration with Microsoft Research Redmond/Asia, and Students@NCSU ASE Group

ICSE Papers: Industry vs. Academia

OSDI 2008 26% vs. xSE ?%

Developers, Programmers, Architects Among All Attendees

average
AC 70%
IND 30%

Industry
Academia



ICSE 2009 Keynote

average 1976-1994

AC 56%

IND 44%



ICSM 2011 Keynote

average 1995-2008

AC 83%

IND 17%

The Role of Creativity: Vendor's View

-SCM Impact Study Findings

- Vendors tend to consider that research impact is restricted to...
 - algorithms (e.g., differencing)
 - pieces of reusable code (e.g., RCS)
- and not...
 - concepts (e.g., hierarchical workspaces)
 - architectures (e.g., peer-to-peer repositories)
- which are often seen as “**engineering common sense**”

The Role of Creativity: Researcher's View

-SCM Impact Study Findings

- Researchers tend to consider that...
 - precedence
 - concepts
 - prototypes
- are sufficient as impact and ignore...
 - efficiency
 - usability
 - reliability
- dismissing them as “**engineering common sense**”

Both are Right and Both are Wrong

-SCM Impact Study Findings

- A good idea is had more than once
- Vendors have disincentives for distributing credit for ideas
- Researchers have incentives for claiming credit for ideas
- Research and productization both require **engineering creativity**

What Make Tech Transfer Difficult?

- Scalability
- Complexity
- Applicability
- Usability (human in the loop)
- Cost-Benefit Analysis

Scalability

- Academia
 - Rarely ask “When scale is up, will my solution still work?”
 - Tend to focus on small or toy scale problems
- Real-world (e.g., search engine, code analysis, ...)
 - Often demand a scalable solution
- Ideal: sophisticated and scalable solution
 - But in practice, simple solution tends to be scalable (performance, maintenance, ...)
 - Academia tend to value sophistication > simplicity
- Ex: Test prioritization@Microsoft [ISSTA 2002], Klee [OSDI 2008]

Complexity

■ Academia

- Tend to make assumptions to simplify problems, or one at a time (indeed relaxing assumptions over time)
- May not be able to assess the relevance/feasibility of assumptions in practice; not consult/work w/ industry

■ Real-world

- Often has high complexity, violating these assumptions

■ Example: OO Unit Test Generation

- Isolated simple classes → Isolated complex data structures → Real world classes as focused by our recent work [ESEC/FSE 2009, OOPSLA 2011]

Applicability

■ Academia

- Tend to focus on a solution optimized for one of many situations (likely worse for others) vs. comprehensive solution
- May not enable to tell ahead of time whether a given case would fall into applicable scope of the solution

■ Real-world

- Need a comprehensive solution that would work generally (at least not compromising too much other situations)

■ Examples

- Integration of our Fitnex in Pex [DSN 2009]
- Coverity [CACM 2010] vs. MSRA XIAO/PatternInsight
- Industry adoption of open source tools

Usability

■ Academia

- Tend to leave human out of loop (involving human makes evaluations difficult to conduct or write)
- Tend not to spend effort on improving tool usability
 - tool usability would be valued more in HCI than in SE
 - too much to include both the approach/tool itself and usability/its evaluation in a single paper

■ Real-world

- Often has human in the loop (familiar IDE integration, social effect, lack of expertise/willingness to write specs,...)

■ Examples

- Agitar [ISSTA 2006] vs. Daikon [TSE 2001]
- Debugging user study [ISSTA 2011]

"Are Automated Debugging [Research] Techniques Actually Helping Programmers?"

- 50 years of automated debugging research
 - N papers → only 5 evaluated with actual programmers

“ Programmers have been waiting a long time for usable automated debugging tools, and we have already gone a long way from the early days of debugging. We believe that, to further advance the state of the art in this area, we must steer research towards more promising directions that take into account the way programmers actually debug in real scenarios. ”



Cost-Benefit Analysis

■ Academia

- Tend to focus on one or a few dimensions of measurement (e.g., analysis cost, precision and/or recall)

■ Real-world

- Consider many dimensions of measurement
 - Cost, e.g., human cost
 - Benefit, e.g., bug severity

■ Example

- FindBugs experience at Google [ISSTA 2009]

Evaluation of Design/PL

“Research in Programming Languages” -Lopes

- “Since the 90s, a considerable percentage of new languages that ended up being very popular were designed by lone programmers, some of them kids with no research inclination, some as a side hobby, and without any grand goal other than either making some routine activities easier or for plain hacking fun.” – PHP, JavaScript, Python, Ruby
- *“one striking commonality in all modern programming languages, especially the popular ones, is how little innovation there is in them!”*
- *“reverse the trend of placing software research under the auspices of science and engineering [alone]”*

<http://tagide.com/blog/2012/03/research-in-programming-languages/>

Why Do Some Programming Languages Live and Others Die?

Wired.com

- Part of the problem is that language designers don't always have practical objectives. There's a tendency in academics of trying to solve a problem when no one actually ever had that problem.
- Academics are so often determined to build a language that stands out from the crowd, without thinking about what's needed to actually make it useful.
 - Sometimes designers fail with the simplest of things, like documentation for their language.
 - Sometimes designers keep adding new features to a language and effectively overload the engineers who are trying to use it.

Suggestions

- Value engineering creativity
- Find killer apps, e.g.,
 - MSR SLAM: Device driver verification
 - MSR Sage: Security testing of binaries
 - PatternInsight/MSRA Xiao: Known-bug detection
- Engage practitioners
 - Get research problems from real practice
 - Get feedback from real practice
 - Collaborate across disciplines
 - Collaborate with industry

Industry Academia Collaboration

- Academia (research recognitions, e.g., papers) vs. Industry (company revenues)
- Academia (research innovations) vs. Industry (likely involving engineering efforts)
- Academia (long-term/fundamental research) vs. Industry (short-term research or work)
- ...
- Industry: problems, infrastructures, data, evaluation testbeds, ...
- Academia: educating students, ...

Personal Interactions with Industry

- Play Around Industrial Tool
 - Parasoft Jtest + Daikon [ASE 03] concurrently with Agitar
 - Parasoft Jtest → Rostra [ASE 04]
- Play Within Industrial Tool
 - Microsoft Research Pex → Fitnexus [DSN 09]
- Advise Industrial Tool Developers
 - Microsoft Research Pex For Fun → [CSE&T 11 Tut]
- Engage Practitioners (indirectly)
 - Microsoft Research Asia Software Analytics Group, e.g., StackMine [ICSE 12]
- Collaborate with Government Agencies
 - FDA, NIST Access Control Policy Tool (ACPT)

Parasoft Jtest

Jolt Awards for Excellence



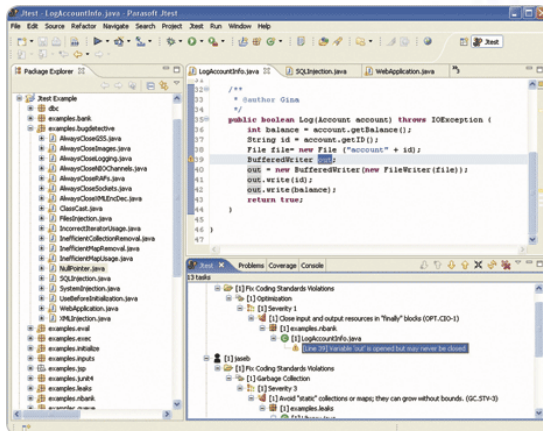
Contact us | Blog

Capabilities Products Solutions Compliance Services

Jtest was recognized with numerous awards, adopted by thousands of development teams worldwide

— businesswire.com

Jtest®



Review, Uni

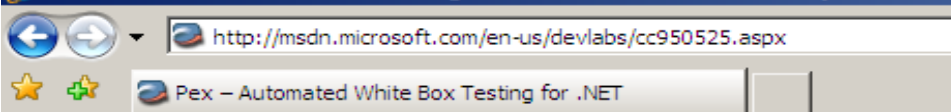
The contributed Rostra approach [ASE 2004] identified 90% tests generated by Parasoft Jtest 4.5 to be redundant. Parasoft fixed issue in later versions after seeing our results

Microsoft Research *Pex*

Incubation Project for Visual Studio



Pex – Automated White Box Testing for .NET - Windows Internet Explorer



msdn

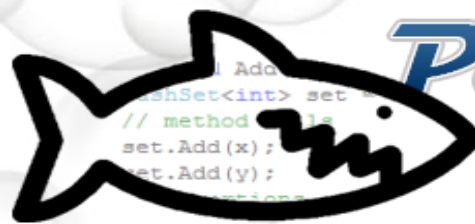
The contributed Fitnexus search strategy [DSN 2009] included in Pex releases since Sept. 2008

Download counts (20 months)
(Feb. 2008 - Oct. 2009)

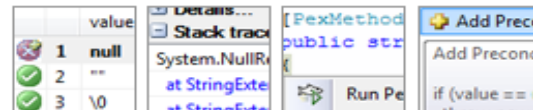
Academic: **17,366**

Devlabs: **13,022**

Total: 30,388



Automated White Box Testing for .NET



About Pex – Automated White Box Testing for .NET see all DevLabs projects...

Pex (Program EXploration) produces a traditional unit test suite with high code coverage. A parameterized unit test is simply a method that takes parameters, calls the code under test, and states assertions. Given a parameterized unit test written in a .NET language, Pex automatically produces a small unit test suite with high code and assertion coverage. To do so, Pex performs a systematic white box program analysis.

<http://research.microsoft.com/projects/pex/>


Use new test cases with different behavior.

Play with Pex, stress it, evaluate it, and tell us what you think.

Microsoft Research *Pex for Fun*

Teaching and Learning CS via Social Gaming

← → www.pexforfun.com ☆ ↻ 🇬🇧 🔍

Curious? Learn More! → **Pex**  Coding Duel *for fun* My Duels ▾ | Settings ▾ | Sign In

Random Puzzle Learn New C# Visual Basic F#

930,875 clicked 'Ask Pex!'

This puzzle is an interactive Coding Duel already won this Duel 305 times! [Help](#) ation? Other people have

```
using System;

public class Program {
    public static int Puzzle(int x) {
        // Can you write code to solve the puzzle?
        return x;
    }
}
```

The contributed concept of **Coding Duel games** as major game type of Pex for Fun since Summer 2010

Ask Pex!

Microsoft Research Asia

Software Analytics

Microsoft
Research
微软亚洲研究院

Microsoft
Research

Search Microsoft Research

Home

Our Research

Connections

Worldwide Labs

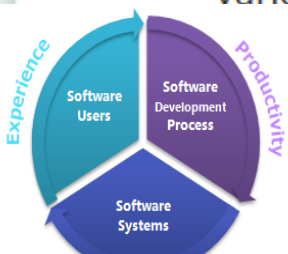
Research Areas

Research Groups

Home > Groups > Software Analytics

Software Analytics

A huge wealth of various data exists in software lifecycle, including source code, feature execution traces/logs, and real-world user feedback, etc. Data plays an essential role. Hidden in the data is information about the quality of software and services as well as various analytical and computing technologies, such as pattern recognition, machine learning, etc.



Information Visualization

Analysis Algorithms

Large-scale Computing

Recent and ongoing work (e.g., StackMine [ICSE 12b]) with successful technology transfer in collaboration with Microsoft Research Asia

Software Analytics as a Learning Case in Practice: Approaches and Experiences

Dongmei Zhang¹, Yingnong Dang¹, Jian-Guang Lou¹, Shi Han¹, Haidong Zhang¹, Tao Xie²

¹Microsoft Research Asia, Beijing, China

²North Carolina State University, Raleigh, NC, USA

{dongmei.zhang, yingnong.dang, jian-guang.lou, shi.han, haidong.zhang}@microsoft.com, xie@csc.ncsu.edu

<http://research.microsoft.com/groups/sa/>

Government Agencies

NIST & FDA



Jointly-developed ACPT
(Access Control Policy Tool)
beta release being beta-tested
in several dozens of
organizations

<http://csrc.nist.gov/groups/SNS/acpt/>



Test a point-of-care assistant
medical device [ASE 10] and
mine FDA incident reports

Summary

- Status of SE research community (e.g., ICSE)
- SIGSOFT Impact project findings
- Challenges for technology transfer
- Suggestions for technology transfer

Thank you!

Questions ?



<https://sites.google.com/site/asergroup/>

Automated
Software Research
Group
Engineering@NCSU